Syntactic Characterisations of Polynomial-Time Optimisation Classes (Syntactic Characterizations of Polynomial-Time Optimization Classes)

P. Manyem*

Abstract

In Descriptive Complexity, there is a vast amount of literature on decision problems, and their classes such as **P, NP, L and NL**. However, research on the descriptive complexity of optimisation problems has been limited. Optimisation problems corresponding to the **NP** class have been characterised in terms of logic expressions by Papadimitriou and Yannakakis, Panconesi and Ranjan, Kolaitis and Thakur, Khanna et al, and by Zimand. Grädel characterised the polynomial class **P** of decision problems. In this paper, we attempt to characterise the optimisation versions of **P** via expressions in second order logic, many of them using universal Horn formulae with successor relations. The polynomially bound versions of maximisation (maximization) and minimisation (minimization) problems are treated first, and then the maximisation problems in the "not necessarily polynomially bound" class.

1 Introduction

Though there has been abundant research in Descriptive Complexity since Fagin's 1974 theorem [Fag74] (which captures the class NP as the set of properties that can be represented in existential second order logic), the application of this area to approximation complexity has been limited. Approximation complexity measures how well an NP-hard optimisation problem can be approximated, or how far is the value of a (possible) heuristic solution from that of an optimal solution.

A few attempts to characterise approximation classes in terms of logic are: Papadimitriou and Yannakakis in 1991 [PY91], Panconesi and Ranjan in 1993 [PR93], Kolaitis and Thakur in 1994 and 1995 [KT94, KT95], and Khanna et al in 1998 [KMSV98].

The approximation complexity of a problem P is usually measured by the *approximation ratio* that a heuristic H for P can guarantee, over all instances of P. The approximation ratio $R_H(I)$ obtained by H for a given instance I of P is given by

$$R_H(I) = \frac{\text{value obtained by } H \text{ on } I}{\text{value of an optimal solution for } I} \tag{1}$$

---

*Centre for Informatics and Applied Optimisation, School of IT and Mathematical Sciences, University of Ballarat, Mount Helen, VIC 3350, Australia. mailto: p.manyem@ballarat.edu.au

In [KT94, PR93, PY91], the authors characterise approximation hardness in terms of quantifier complexity — the number and types of quantifiers that appear at the beginning of a second-order formula in prenex normal form (PNF). For a formula in PNF, all quantifiers appear at the beginning, followed by a quantifier-free formula.

1.1 Contributions in this paper

In this paper, we first present a logical representation of a subclass of $\mathbf{P}'$ — $\mathbf{P}'$ is the class of optimisation problems that can be solved to optimality within polynomial time[1]. The class of *decision problems* corresponding to $\mathbf{P}'$ is $\mathbf{P}$. The particular subclass $\mathbf{Q}'$ (of $\mathbf{P}'$) that we focus on includes only *polynomially bound optimisation problems*, defined below in Definition 1. In particular, we

- provide syntactic characterisations for both maximisation and minimisation problems in $\mathbf{Q}'$,

- give examples of characterisations (MAXFLOW$_{PB}$ for maximisation and SHORTEST PATH$_{PB}$ minimisation),

- show that MAXFLOW$_{PB}$ is complete for the maximisation subclass of $\mathbf{Q}'$,

- present characterisations for maximisation problems in $\mathbf{P}'$ (defined in Table 1 — problems not necessarily polynomially bound), as well as an example for a problem in this class (MAXIMUM MATCHING). This is the most significant contribution in this paper, and is a considerable departure from the treatment in [Zim98]. Whereas that paper studied problems in $\mathbf{N}$ in general, we study maximisation problems in $\mathbf{P}'$.

The syntactic characterisation of $\mathbf{P}$ is given below in Theorem 1, as shown in Grädel [E. 91].

1.2 Notation and Definitions

All notation is defined in Table 1, as a one-stop reference point. For the same reason, all definitions are provided below in this section.

Definition 1. *An optimisation problem Q' is said to be **polynomially bound** if the value of an optimal solution to every instance I of Q' is bound by a polynomial in the size of I. In other words, there exists a polynomial p such that*

$$opt_{Q'}(I) \leq p(|I|), \tag{2}$$

for every instance I of Q'. The class of all such problems is $\mathbf{Q}'$.

---

[1]Strictly speaking, in Turing machine terminology, $\mathbf{P}'$ is the set of languages where, if an instance I of an optimisation problem $P \in \mathbf{P}'$ is encoded as an input string x in some alphabet Σ, a deterministic Turing machine will compute the optimal solution (which is again a string) within $\Theta(|x|^k)$ steps, where k is some constant and $|x|$ is the length of the input string.

σ	vocabulary
A	a structure defined over σ (captures an instance of an optimisation problem)
η	a quantifier-free first order formula, and a conjunction of *Horn* clauses at the same time. (Recall that a Horn clause contains at most one positive literal.)
x	an $m-$tuple of first order variables
S	a sequence of second-order variables (predicate symbols) (captures a solution to the optimisation problem)
P	computational class of *decision* problems, decidable in polynomial time by a deterministic Turing machine
P′	class of *optimisation* problems corresponding to **P** (also called **P**−optimisation problems)
Q′	$\mathbf{Q'} \subseteq \mathbf{P'}$, and $\mathbf{Q'}$ only contains *polynomially bound* optimisation problems (see Definition 1)
N	class of *optimisation* problems whose decision versions are in **NP**
N′	$\mathbf{N'} \subseteq \mathbf{N}$, and $\mathbf{N'}$ only contains *polynomially bound* optimisation problems
ESO	Existential Second Order Logic
PNF	Prenex Normal Form

Table 1: Notation

Definition 2. *First order logic consists of a vocabulary (alias signature) σ, and models (alias structures) defined on the vocabulary. In its simplest form, a vocabulary consists of a set of variables, and a set of relation symbols $R_j (1 \leq j \leq J)$, each of arity r_j. A model M consists of a universe U whose elements are the values that variables can take — M also instantiates each relation symbol $R_j \in \sigma$ with tuples from $U^{(r_j)}$. For example, a model $\mathbf{G}$ in graph theory may have the set of vertices $G = \{1, 2, \cdots 10\}$ as its universe (assuming that the graph has 10 vertices), and a single binary relation E where $E(i, j)$ is true iff (i, j) is an edge in the graph $\mathbf{G}$. A model represents an instance of an optimisation problem.*

Definition 3. *A $\mathbf{\Pi_1}$ ($\mathbf{\Sigma_1}$) **first order formula in PNF** only has universal (existential) quantifiers, quantified over first order variables.*

Definition 4. *An **existential second-order (ESO) Horn** expression is of the form $\exists \mathbf{S}\psi$, where ψ is a first order formula, and $\mathbf{S} = (S_1, \cdots S_p)$ is a sequence of predicate symbols not in the vocabulary of ψ. The formula ψ can be written in Π_1 form as*

$$\psi = \forall x_1 \forall x_2 \cdots \forall x_k \eta = \forall \mathbf{x} \ \eta. \tag{3}$$

where η is a conjunction of Horn clauses (η is, of course, quantifier-free), and x_i $(1 \leq i \leq k)$ are first order variables. Each clause in η contains at most one positive occurrence of any of the second order predicates S_i $(1 \leq i \leq p)$.

Definition 5. *A $\mathbf{\Pi_2}$ ($\mathbf{\Sigma_2}$) **formula in prenex normal form (PNF)** can be written as follows:*

$$\phi = \forall x_1 \cdots \forall x_a \ \exists y_1 \cdots \exists y_b \ \eta \quad (\phi = \exists y_1 \cdots \exists y_b \ \forall x_1 \cdots \forall x_a \ \eta), \tag{4}$$

where η is quantifier-free, $a, b \geq 1$, and the x's and y's are first-order variables.

The following theorem is due to Grädel [E. 91] — this is the polynomial-time counterpart of Fagin's theorem [Fag74] which characterised the class **NP**:

Theorem 1. *For any ESO Horn expression as defined in Definition 4, the corresponding decision problem is in $\mathbf{P}$.*

The converse is also true — if a problem P is in $\mathbf{P}$, then it can be expressed in ESO Horn form — but only if a successor relation is allowed to be included in the vocabulary of the first-order formula ψ.

2   Polynomially Bound Optimisation Problems Q$'$

Optimisation problems corresponding to P. We assume that for a maximisation (or a minimisation) problem Q' in the class $\mathbf{Q}'$ (corresponding to the class $\mathbf{P}$ of decision problems), the following can be computed in polynomial time deterministically: (a) The value of the objective function $f(\mathbf{A}, \mathbf{S})$ to a solution $\mathbf{S}$ of an instance $\mathbf{A}$, and (b) Whether a solution $\mathbf{S}$ is a feasible solution to an instance $\mathbf{A}$.

We will study maximisation problems first, and then the minimisation problems.

2.1 Polynomially Bound P-Maximisation Problems

For maximisation problems in $\mathbf{N}'$ (see Table 1 for a definition of $\mathbf{N}'$), Kolaitis and Thakur [KT94] proved the following:

Theorem 2. *A maximisation problem $Q \in \mathbf{N}'$ if and only if there exists a Π_2 first order formula $\phi(\mathbf{w}, \mathbf{S})$ with predicate symbols from the vocabulary σ (of ϕ) and the sequence $\mathbf{S}$, such that for every instance $\mathbf{A}$ of Q, the optimal solution value is given by*

$$opt_Q(\mathbf{A}) = \max_{\mathbf{S}} |\{\mathbf{w} : (\mathbf{A}, \mathbf{S}) \models \phi(\mathbf{w}, \mathbf{S})\}|. \tag{5}$$

In other words, polynomially bound NP-maximisation problems fall in what is called the MAX Π_2 class. We can show a similar result for the polynomial-time counterpart of $\mathbf{N}'$, that is, maximisation problems in $\mathbf{Q}'$:

Theorem 3. *Let $\mathbf{A}$ be a structure (instance) defined over σ. The value of an optimal solution to an instance $\mathbf{A}$ of a maximisation problem Q' can be represented by*

$$opt_{Q'}(\mathbf{A}) = \max_{\mathbf{S}} |\{\mathbf{w} : (\mathbf{A}, \mathbf{S}) \models \forall \mathbf{x} \ \eta(\mathbf{w}, \mathbf{x}, \mathbf{S})\}| \tag{6}$$

if $Q' \in \mathbf{Q}'$, where $\mathbf{x}$, $\mathbf{A}$, $\mathbf{S}$ and η are defined in Table 1.

Proof. Let Q and Q' be the decision and optimisation versions respectively.

We first show that Q' is polynomially bound. For this, the number of tuples $\mathbf{w}$ in an optimal solution $\mathcal{S}^*$ should be polynomial in $|A|$, the size of the universe of $\mathbf{A}$. It suffices to show this for any solution $\mathcal{S}$. Recall from Table 1 that the sequence $\mathbf{S}$ of predicates captures a (corresponding) solution $\mathcal{S}$ to the optimisation problem.

Suppose $\mathbf{w}$ is a $\mathcal{R}-$dimensional tuple. For a given universe A of $\mathbf{A}$, the number of possible tuples is $|A|^{\mathcal{R}}$ — this is true for any solution $\mathcal{S}$ including the optimal one. Hence Q' is polynomially bound.

To complete the proof, we should show that if $Q' \in \mathbf{Q}'$, then the optimal solution value to an instance $\mathbf{A}$ of Q' can be represented by equation (6).

Refer to Grädel's theorem (Theorem 1). The decision problem Q can be written as an ESO Horn expression $\exists \mathbf{S} \psi$, except that now, ψ should include a successor relation in its vocabulary, in addition to being a Horn first order formula. Problem Q can be posed as: Given an instance (a finite structure) $\mathbf{A}$, is there a feasible solution $\mathbf{S}$ such that $f(\mathbf{A}, \mathbf{S}) \geq K$, where K is a certain integer ?
(Here f is the value of the objective function to solution $\mathbf{S}$ for the optimisation problem. Assume that we deal only with problems with integer-valued objective functions.)

A feasible solution $\mathbf{S}$ could consist of several relations $S_1, S_2, \cdots S_p$ of arities $r_1, r_2, \cdots r_p$. The formula ψ should be able to express $f(\mathbf{A}, \mathbf{S}) \geq K$ — however, this is insufficient. Given a solution $\mathbf{S}$, we know that all feasibility conditions, including $f(\mathbf{A}, \mathbf{S}) \geq K$, can be checked in polynomial time deterministically for all problems in $\mathbf{NP}$. What distinguishes $\mathbf{P}$ from $\mathbf{NP}$ is the fact that for optimisation problems corresponding to $\mathbf{P}$, the optimal solution value $opt_{Q'}(\mathbf{A})$ can be computed in polynomial time deterministically, whereas for optimisation

problems corresponding to **NP**, we only know that this value can be computed in polynomial time non-deterministically.

Hence this condition should be modified to $g(\mathbf{A}) = opt_{\mathbf{S}} f(\mathbf{A}, \mathbf{S}) \geq K$, where $g(\mathbf{A})$ is the optimal solution value to instance $\mathbf{A}$ over all solutions $\mathbf{S}$. Each of the $g(\mathbf{A})$ number of entities can be considered to be a tuple $\mathbf{w_i}$, and thus we need at least K such tuples. These tuples, at least K in number, can be defined to form a new relation F (on the universe A of $\mathbf{A}$) of arity k. Thus we want $|F|$, the number of tuples $\mathbf{w}$ that satisfy $F(\mathbf{w})$, to be at least K.

Digression to discuss arity k. As examples, setting $k = 2$ will suffice for the LONGEST PATH problem where the number of arcs in a path is to be maximised, and $k = 1$ in a MAXSAT problem where the number of satisfying clauses is to be maximised. However, this can handle only up to very small values of the objective function. In the LONGEST PATH case, we can only count $|A|^2$ tuples at most (where $|A|$ is the number of vertices in the graph). However, if arc lengths are higher than one, but still polynomially bound in $|A|$, the length of the longest path — though still polynomially bound in $|A|$ — could be well above $|A|^2$, and this length cannot be handled by an arity of $k = 2$ — a higher arity is required. Hence it would be safest to increase the arity to $\mathcal{R}$, since $|A|^{\mathcal{R}}$ is the upper bound on the objective function value. A similar argument applies to the weighted MAXSAT problem with polynomially bound weights — and to all polynomially bound NP-maximisation problems in general.

Recall from Theorem 1 that ψ is in Π_1 form, where the quantifier-free part of ψ is a conjunction of Horn clauses, each of which contains at most one positive occurrence of any^2 of the relation symbols S_i. Hence ψ can be written as $\forall x_1, \cdots \forall x_m \hat{\eta}$, where $\hat{\eta}$ is an expression consisting of variables $x_1, \cdots x_m$, all predicates from $\mathbf{S}$, and the the relation F. That is,

$$\psi = \forall x_1 \cdots \forall x_m \ \hat{\eta}(x_1, \cdots, x_m, F, \mathbf{S}) = \forall \mathbf{x} \ \hat{\eta}(\mathbf{x}, F, \mathbf{S}), \tag{7}$$

where $\hat{\eta}$ is a conjunction of Horn clauses ($\hat{\eta}$ captures the feasibility of solution $\mathbf{S}$) and $\mathbf{x} = (x_1, \cdots, x_m)$. Note that $\hat{\eta}$ needs to capture two types of conditions (an example with such conditions is provided in the next section):

(a) *Global* conditions (those that apply over all $\mathbf{w}$ tuples): Such conditions express the fact that the solution ($\mathbf{S}$, F) as a whole is a feasible solution to $\mathbf{A}$. One such condition is $|F| \geq K$ mentioned above. And

(b) *Local* conditions: The ones that are specific to a given $\mathbf{w}$ — if $F(\mathbf{w})$ is true, that is.

Thus $\hat{\eta}$ is a conjunction of these two types of conditions. The global conditions[3] can be written as $\hat{\eta}_1$, and the local conditions as $\forall \mathbf{w} \ F(\mathbf{w}) \longrightarrow \hat{\eta}_2(\mathbf{x}, \mathbf{w}, \mathbf{S})$. So Q, the decision problem, can be written as $\exists \mathbf{S} \ \exists F \ \forall \mathbf{x} \ \hat{\eta}_1 \wedge [\forall \mathbf{w} \ F(\mathbf{w}) \longrightarrow \hat{\eta}_2(\mathbf{x}, \mathbf{w}, \mathbf{S})]$. In prenex normal form,

$$Q \ \equiv \ \exists \mathbf{S} \ \exists F \ \forall \mathbf{w} \ \forall \mathbf{x} \ \hat{\eta}_1 \wedge [F(\mathbf{w}) \longrightarrow \hat{\eta}_2(\mathbf{x}, \mathbf{w}, \mathbf{S})]$$

$$\equiv \ \exists \mathbf{S} \ \exists F \ \forall \mathbf{w} \ \forall \mathbf{x} \ \hat{\eta}_1 \wedge [\neg F(\mathbf{w}) \vee \hat{\eta}_2(\mathbf{x}, \mathbf{w}, \mathbf{S})] \tag{8}$$

---

[2]For example, if S_1 and S_2 are second order predicates, then a Horn clause cannot contain both S_1 and S_2 as positive literals.

[3]Observe that $\hat{\eta}_1$ captures the cardinality condition $|F| \geq K$. To represent this, we can define a first-order relation G of arity k over the universe A of $\mathbf{A}$ such that $|G| = K$ and $F(\mathbf{w})$ is true whenever $G(\mathbf{w})$ is. Then we need to represent the fact that $|F| \geq |G|$, which can be characterised as $\forall \mathbf{w} \ G(\mathbf{w}) \longrightarrow F(\mathbf{w})$.

If $\hat{\eta}_1$ and $\hat{\eta}_2$ are each a conjunction of Horn clauses, then so is the formula in (8). If we let $\eta(\mathbf{x}, \mathbf{w}, \mathbf{S}, F) = \hat{\eta}_1 \wedge [\neg F(\mathbf{w}) \vee \hat{\eta}_2(\mathbf{x}, \mathbf{w}, \mathbf{S})]$, then (8) can be rewritten in ESO Horn Π_1 form as

$$Q \equiv \exists \mathbf{S} \; \exists F \; \forall \mathbf{w} \; \forall \mathbf{x} \; [\eta(\mathbf{x}, \mathbf{w}, \mathbf{S}, F)]. \tag{9}$$

To express the optimal solution value for Q', we maximise over all feasible solutions $\mathbf{S}$ — and for each solution, count the number of $\mathbf{w}$ tuples for which the relation $F(\mathbf{w})$ and $\psi(\mathbf{w}, \mathbf{S})$ hold[4]:

$$opt_{Q'}(\mathbf{A}) = \max_{\mathbf{S}, F} |\{\mathbf{w} : (\mathbf{A}, \mathbf{S}, F) \models [\forall \mathbf{x} \; \hat{\eta}(\mathbf{x}, \mathbf{w}, \mathbf{S})] \wedge F(\mathbf{w})\}|. \tag{10}$$

In (10), $\forall \mathbf{x} \; \hat{\eta}(\mathbf{x}, \mathbf{w}, \mathbf{S})$ represents the feasibility of the given instance $\mathbf{A}$.

If $\mathbf{S}$ and F can be represented by a single sequence of relations $\mathbf{T} = (S_1, S_2, \cdots S_p, F)$, the optimal solution value can be expressed as

$$opt_{Q'}(\mathbf{A}) = \max_{\mathbf{T}} |\{\mathbf{w} : (\mathbf{A}, \mathbf{T}) \models \forall \mathbf{x} \; \eta(\mathbf{x}, \mathbf{w}, \mathbf{T})\}| \tag{11}$$

where $\eta(\mathbf{x}, \mathbf{w}, \mathbf{T}) = \hat{\eta}(\mathbf{x}, \mathbf{w}, \mathbf{S}) \wedge F(\mathbf{w})$. ($\hat{\eta}$ and η are quantifier-free.) Since $\hat{\eta}$ is Horn, so is η.

Hence the proof. $\square$

2.1.1 Example: Polynomially Bound Maximum Flow (Unit Capacities)

In this section, we will see how the MAXFLOW problem with unit capacities can be expressed in ESO, in Π_1 form. Given a source s and a sink t, and a network G containing directed edges, we want to find the maximum flow that can be sent through the network from s to t. Essentially we seek the maximum number of edge-disjoint paths from s to t. Call this (polynomially bound) problem MAXFLOW_{PB}.

We want to determine the maximum number of vertices w (one dimensional tuples) to which there is a flow from s, along the edge (s, w) (if such an edge exists) — this will give us the value of the maximum flow from s to t. Every $s - t$ edge-disjoint path can be considered as a partial order on the set of vertices. We will use ideas similar to those used in the expression for REACHABILITY [Pap94].

To represent the partial orders, introduce a second-order ternary predicate $P(x, y, w)$ which holds iff $x \neq y$ and there is an edge-disjoint path from s to w to x to y, in the feasible solution — the path from s to w, is of course, just a single edge. (The "main arguments" for P are x and y — w is just an additional reference.) Thus we seek the maximum number of w's such that $P(w, t, w)$ is true. The following expressions capture the properties of a feasible solution.

(1) If $P(x_1, x_2, w)$ holds, then so does $G(s, w)$ — that is, the edge (s, w) is defined in G:

$$\begin{aligned} \phi_1 &\equiv \forall x_1 \forall x_2 \forall w \; P(x_1, x_2, w) \longrightarrow G(s, w) \\ &\equiv \forall x_1 \forall x_2 \forall w \; \neg P(x_1, x_2, w) \vee G(s, w). \end{aligned} \tag{12}$$

---

[4] Out of the four possible cases (i) $(\forall \mathbf{x} \hat{\eta}) \wedge F(\mathbf{w})$, (ii) $(\forall \mathbf{x} \hat{\eta}) \wedge \neg F(\mathbf{w})$, (iii) $(\neg \forall \mathbf{x} \hat{\eta}) \wedge F(\mathbf{w})$, and (iv) $(\neg \forall \mathbf{x} \hat{\eta}) \wedge \neg F(\mathbf{w})$, cases (ii) and (iv) must be disregarded since F is false. Case (iii) should also be disregarded since it violates the feasibility condition $\forall \mathbf{x} \hat{\eta}$.

(2) An edge (i, j) can be a part of only one $s - t$ disjoint path (equivalently, only one $w - t$ edge disjoint path):

$$
\begin{aligned}
\phi_2 &\equiv \forall i \forall j \forall w_1 \forall w_2 \ P(i, j, w_1) \wedge P(i, j, w_2) \wedge G(i, j) \longrightarrow (w_1 = w_2) \\
&\equiv \forall i \forall j \forall w_1 \forall w_2 \ \neg P(i, j, w_1) \vee \neg P(i, j, w_2) \vee \neg G(i, j) \vee (w_1 = w_2).
\end{aligned}
\tag{13}
$$

(3) P is non-reflexive:

$$
\phi_3 \equiv \forall y_1 \forall y_2 \ \neg P(y_1, y_1, y_2).
\tag{14}
$$

(4) P is transitive:

$$
\begin{aligned}
\phi_4 &\equiv \forall u_1 \forall u_2 \forall u_3 \forall w_3 \ P(u_1, u_2, w_3) \wedge P(u_2, u_3, w_3) \longrightarrow P(u_1, u_3, w_3). \\
&\equiv \forall u_1 \forall u_2 \forall u_3 \forall w_3 \ \neg P(u_1, u_2, w_3) \wedge \neg P(u_2, u_3, w_3) \vee P(u_1, u_3, w_3).
\end{aligned}
\tag{15}
$$

(5) And finally, any two adjacent vertices in P should also be adjacent in G:

$$
\begin{aligned}
\phi_5 &\equiv \forall z_1 \forall z_2 \forall w_4 \ P(z_1, z_2, w_4) \wedge \ \forall z_3 \ \neg[P(z_1, z_3, w_4) \wedge P(z_3, z_2, w_4)] \longrightarrow G(z_1, z_2) \\
&\equiv \forall z_1 \forall z_2 \forall z_3 \forall w_4 \ P(z_1, z_2, w_4) \wedge \neg[P(z_1, z_3, w_4) \wedge P(z_3, z_2, w_4)] \longrightarrow G(z_1, z_2) \\
&\equiv \forall z_1 \forall z_2 \forall z_3 \forall w_4 \ \neg P(z_1, z_2, w_4) \vee [P(z_1, z_3, w_4) \wedge P(z_3, z_2, w_4)] \vee G(z_1, z_2) \\
&\equiv \forall z_1 \forall z_2 \forall z_3 \forall w_4 \ [\neg P(z_1, z_2, w_4) \vee P(z_1, z_3, w_4) \vee G(z_1, z_2)] \\
&\qquad \wedge \ [\neg P(z_1, z_2, w_4) \vee P(z_3, z_2, w_4) \vee G(z_1, z_2)].
\end{aligned}
\tag{16}
$$

Let $\Phi = \bigwedge_{i=1}^{5} \phi_i$.

Observe that each ϕ_i $(1 \leq i \leq 5)$ is a Π_1 Horn formula, as required by Theorem 3. The optimal solution value to the given instance (network G, represented by a structure $\mathbf{A}$), is given by

$$
opt_Q(\mathbf{A}) = \max_P |\{w : (\mathbf{A}, P) \models \ P(w, t, w) \wedge \Phi\}|.
\tag{17}
$$

Discussion. In most such expressions as above, there are two types of conditions to be expressed: (1) *Global* conditions (those that apply over all $\mathbf{w}$ tuples), such as the expression Φ above and (2) *Local* conditions (the ones that are specific to a given $\mathbf{w}$), such as $P(w, t, w)$ above. The first (second) set of conditions correspond to *constraints* (*objective function*) in a classical mathematical programming framework.

It is clear that MAXFLOW$_{PB}$ can be expressed with neither Σ_0 nor Σ_1 formulae. In particular, without universal quantifiers, none of the five properties — expressions (12) to (16) — can be expressed independently of the size of the instance.

Consider property ϕ_2, for instance. Using only existential quantifiers, one should enumerate the property individually for each edge. However, this will make the length of ϕ_2 dependent on the number of edges in the graph. Hence we can conclude that

Proposition 4. *The property, that an edge belongs at most one edge-disjoint $s - t$ path in a solution, (and hence the MAXFLOW$_{PB}$ problem) can be expressed with a Π_1 formula, but not with a Σ_1 formula.*

2.1.2 MAXFLOW$_{\mathbf{PB}}$ is Complete for Polynomially Bound Maximisation

We can show that the MAXFLOW$_{PB}$ problem is complete for the class of polynomially bound maximisation problems by reducing an instance I of a general problem Q' in this class to an instance $\mathcal{I}$ of MAXFLOW$_{PB}$. If I is represented by a structure $\mathbf{A}$, the optimal solution value to I is given by Theorem 3:

$$opt_{Q'}(\mathbf{A}) = \max_{\mathbf{S}} |\{\mathbf{w} : (\mathbf{A}, \mathbf{S}) \models \Phi\}|, \quad \Phi = \forall \mathbf{x} \; \eta(\mathbf{w}, \mathbf{x}, \mathbf{S}) \tag{18}$$

if $Q' \in \mathbf{Q}'$, where $\mathbf{x}$, $\mathbf{A}$, $\mathbf{S}$ and η are defined in Table 1. (Recall that η is a conjunction of Horn clauses and quantifier-free, and $\mathbf{S}$ is a sequence of second order predicate symbols.)

Let the arity of $\mathbf{w}$ ($\mathbf{x}$) be k (m) respectively. The different possible $\mathbf{w}$ ($\mathbf{x}$) tuples are $\mathbf{w_i}$, $1 \leq i \leq n^k$ ($\mathbf{x_j}$, $1 \leq j \leq n^m$), where n is the cardinality of the universe A of $\mathbf{A}$. For a given $\mathbf{w_i}$, the expression for Φ in (18) can be rewritten as

$$\Phi(\mathbf{w_i}) = \forall \mathbf{x} \; \eta(\mathbf{w_i}, \mathbf{x}, \mathbf{S}) = \bigwedge_{j=1}^{n^m} \eta(\mathbf{w_i}, \mathbf{x_j}, \mathbf{S}). \tag{19}$$

Instance $\mathcal{I}$ consists of $n^k + 2$ vertices — one for each $\mathbf{w_i}$ tuple, as well as two additional vertices s and t. Add a directed edge with unit capacity from s to each $\mathbf{w_i}$ vertex. Add a directed edge with unit capacity from each $\mathbf{w_i}$ vertex to t iff $\Phi(\mathbf{w_i})$ holds.

The reduction is polynomial time — $O(n^k)$ time to create the vertices, and $O(n^{k+m})$ time to add the edges. It is clear that instance $\mathcal{I}$ of MAXFLOW$_{PB}$ has a maximum flow of α units from source s to sink t iff the optimal solution value to I in (18) is also α.

2.1.3 A Problem in MAX$_{\mathbf{P}}\Sigma_0$?

Within the class $\mathbf{Q}'$ (see Table 1), let us define the class MAX$_P\Sigma_0$ (MAX$_P\Pi_1$) as the class of polynomially bound maximisation problems that can be expressed by a Σ_0 (Π_1) formula.

Kolaitis and Thakur showed that MAX3SAT is in MAX$_{NP}\Sigma_0$ (defined similar to MAX$_P\Sigma_0$), a subset of $\mathbf{N}'$. On the other hand, MAX2SAT is not known to be polynomially solvable [H97], though the decision version, as to whether all clauses are satisfiable, is well-known to be in $\mathbf{P}$ [GJ79]. Results similar to MAX2SAT for both the maximisation and decision versions are also known for HORNSAT (where every clause is required to be a Horn clause) [KKM94].

Towards the goal of obtaining a hierarchy within the polynomially bound P-maximisation class, we need to exhibit a problem in MAX$_P\Sigma_0$. However, we have been unable to find such problem so far.

We conjecture that as long as a successor relationship or a linear ordering on the universe of a structure is necessary, a problem cannot be expressed in MAX$_P\Sigma_0$ (since this will require a Π_1 expression).

2.1.4 Hierarchy Within Maximisation

We state the hierarchy within the polynomially bound maximisation class without a formal proof (since it is clear from the argument below): If a maximisation problem exists in the Σ_0 class (or the Π_0 class), then the Σ_0 class is strictly contained within the Π_1 class.

It is clear that the problem considered in Section 2.1.1, MAXFLOW$_{PB}$, cannot be expressed with a Σ_0 formula. In particular, without quantifiers, none of the five properties in Section 2.1.1 — expressions (12) to (16) — can be expressed.

From Section 2.1.2, clearly MAXFLOW$_{PB}$ serves as a complete problem for the MAX$_P\Pi_1$ class. It would be desirable to obtain a complete problem for the MAX$_P\Sigma_0$ class. An interesting observation is that the decision version of the weighted MAXFLOW problem (where arc capacity can be any non-negative integer) is complete for the class **P** [GHR95, Imm99].

Another question to be answered is, is there a class MAX$_P\Sigma_1$ which is between MAX$_P\Pi_1$ and MAX$_P\Sigma_0$?

2.2 Polynomially Bound P-Minimisation Problems

For minimisation problems in $\mathbf{N}'$, Kolaitis and Thakur [KT94] proved the following (see Table 1 for a definition of $\mathbf{N}'$ and Definition 5 regarding Σ_2 formulae):

Theorem 5. *A minimisation problem $Q' \in \mathbf{N}'$ if and only if there exists a Σ_2 first order formula $\phi(\mathbf{w}, \mathbf{S})$ with predicate symbols from the vocabulary σ (of ϕ) and the sequence $\mathbf{S}$, such that for every instance $\mathbf{A}$ of Q',*

$$opt_{Q'}(\mathbf{A}) = \min_{\mathbf{S}} | \{\mathbf{w} : (\mathbf{A}, \mathbf{S}) \models \phi(\mathbf{w}, \mathbf{S})\} |. \tag{20}$$

In other words, they showed that all polynomially bound NP-minimisation problems fall in what can be called the MIN Σ_2 class. In the same paper, they also showed that this class is equivalent to the MIN Π_1 class.

A similar result can be shown for minimisation problems in $\mathbf{Q}'$ (the polynomial-time equivalent of $\mathbf{N}'$):

Theorem 6. *Let $\mathbf{A}$ be a structure (instance) defined over σ. If Q' is a minimisation problem in $\mathbf{Q}'$, then the value of an optimal solution to an instance $\mathbf{A}$ of Q' can be represented by*

$$opt_{Q'}(\mathbf{A}) = \min_{\mathbf{S}, F} | \{\mathbf{w} : (\mathbf{A}, \mathbf{S}, F) \models \forall \mathbf{x}\ \tau\} | \tag{21}$$

where $\tau = \eta(\mathbf{w}, \mathbf{x}, \mathcal{S}) \wedge F(\mathbf{w})$, and $\mathbf{x}$, $\mathbf{A}$, $\mathbf{S}$, η are defined as in Table 1. (The symbol F is a $k-ary$ relation defined on the universe $|A|$ of $\mathbf{A}$, since each $\mathbf{w}$ is $k-dimensional$.)

Proof. The proof that Q' is polynomially bound is the same as in Theorem 3.

We start with Grädel's Theorem (Theorem 1). The decision problem can be represented by an ESO Horn expression $\exists \mathbf{S}\psi$ where $\mathbf{S}$ is a sequence of predicate symbols, and ψ is a Π_1 first order Horn expression where a successor relation is included in the vocabulary of ψ.

The analysis for the decision problem Q is similar to the maximisation case, except that one looks for at most K tuples that satisfy the feasibility condition $\eta(\mathbf{x}, \mathbf{W}, \mathbf{S})$[5].

Thus for the minimisation version of the problem, an optimal value to an instance $\mathbf{A}$ can be written as

$$opt_{Q'}(\mathbf{A}) = \min_{\mathbf{S}} |\{\mathbf{w} : (\mathbf{A}, \mathbf{S}) \models \phi(\mathbf{w}, \mathbf{S})\}| \tag{22}$$

where $\phi(\mathbf{w}, \mathbf{S}) = \forall \mathbf{x}\ \eta(\mathbf{x}, \mathbf{w}, \mathbf{S})$.

The $\mathbf{w}$ tuples can be considered as a $k-$ary relation F such that $F(\mathbf{w})$ is true if and only if $\mathbf{w} \in F$. Hence $\phi(\mathbf{w}, \mathbf{S})$ in (22) should be modified to $\forall \mathbf{x}\ \eta(\mathbf{x}, \mathbf{w}, \mathbf{S}) \wedge F(\mathbf{w})$. The number of tuples $|F|$ in F should be minimised.

Again, out of the the four cases (i) $(\forall \mathbf{x}\eta) \wedge F(\mathbf{w})$, (ii) $(\forall \mathbf{x}\eta) \wedge \neg F(\mathbf{w})$, (iii) $(\neg\forall \mathbf{x}\eta) \wedge F(\mathbf{w})$, and (iv) $(\neg\forall \mathbf{x}\eta) \wedge \neg F(\mathbf{w})$, cases (ii) and (iv) should be disregarded since F is false, and (iii) violates the feasibility condition $\forall \mathbf{x}\eta$. This leaves us with the following modification of (22):

$$opt_{Q'}(\mathbf{A}) = \min_{\mathbf{S}, F} |\{\mathbf{w} : (\mathbf{A}, \mathbf{S}) \models (\forall \mathbf{x}\eta) \wedge F(\mathbf{w})\}| \tag{23}$$

Note that $(\forall \mathbf{x}\eta) \wedge F(\mathbf{w}) = \forall \mathbf{x}(F(\mathbf{w}) \wedge \eta)$. Since η is Horn, so is $(F(\mathbf{w}) \wedge \eta)$.

It may appear that the minimisation in (23) will always result in an optimal value $|F|$ of zero, but since the minimum value is taken only over all *feasible* solutions $(\mathbf{S}, F)$, the value obtained in (23) is correct. Hence the proof. (The minimisation over "only feasible solutions $(\mathbf{S}, F)$" needs further illustration and is provided below.) $\square$

Illustration of Minimisation over "only feasible solutions $(\mathbf{S}, F)$". To illustrate this point, consider the SHORTEST PATH problem in Section 2.2.1. We attempt to minimise the number of edges in a path from the source s to the sink t. If we minimise over **any** $(\mathbf{S}, F)$ combination, obviously this minimum number would be zero — however, this would violate the feasibility condition ϕ_1 that there exists a path from s to t. Hence this "zero" solution is obviously infeasible.

Consider another example, MIN SET COVER. We are given a *ground* set X and several subsets $Y_1, Y_2, \cdots, Y_q$ of X. Let $C = \{Y_1, Y_2, \cdots, Y_q\}$. The problem is select a few (minimum number of) subsets Y_i from C, such that the union of the selected subsets is X. We can associate a unary tuple w_i to each Y_i, such that the number of such w tuples in a solution is to be minimised. Let a second order predicate $S(w_i)$ determine if a certain subset Y_i is chosen in a solution S. Obviously if we minimise over *all possible S*, the minimum number of Y_i subsets selected will be zero — but then, such a solution is clearly infeasible, since the union of the selected subsets (zero of them!) is not equal to the ground set X.

Discussion. From Theorems 2-5 and 3-6, the following can be observed in the case of polynomially bound optimisation problems:

While second-order expressions are able to distinguish clearly between NP-maximisation and P-maximisation problems (Π_2 for the former and Horn Π_1 for the latter), the distinction is less clear between NP-minimisation and P-minimisation problems (Π_1 formulae in both cases, the only distinction being the Horn clause requirement in the P-minimisation case).

---

[5]...... which is the same as looking for at least $n^k - K + 1$ tuples that do not satisfy $\eta(\mathbf{x}, \mathbf{W}, \mathbf{S})$.

2.2.1 Example: Shortest Path

We now provide an example of a polynomially bound P-minimisation problem, SHORTEST PATH$_{PB}$. Assume that the edges have unit weight and are directed. The number of edges in the shortest path is to be minimised. The decision version of this problem is easily represented as a Σ_1 formula:

$$\exists x_1 \exists x_2 \cdots \exists x_k \; G(s, x_1) \wedge G(x_1, x_2) \wedge \cdots \wedge G(x_{k-1}, x_k) \wedge G(x_k, t) \tag{24}$$

where s is the origin and t is the destination. The above formula says that there is a path from s to t of length $k + 1$, and it is a Horn formula. ($G(x, y)$ is true if there exists an arc (x, y) in graph G.) We have not used any second-order variables in (24), hence the decision version is *FO (first order) expressible*.

Minimisation version. A shortest path (or any path from origin to destination) represents a partial order P on the universe (the set of vertices) — P is represented by a second order (SO) binary predicate. Another SO binary predicate S chooses which arcs in the network are in the required path. Again, we will use ideas similar to those used for REACHABILITY [Pap94]. The following formulae express the properties of P and S:

$$
\begin{aligned}
\phi_1 &\equiv & P(s, t) \equiv \eta_1 \quad \text{(there exists a path from } s \text{ to } t). \\
\phi_2 &\equiv & \forall x \; \forall y \; \forall z \; \eta_2, \quad \eta_2 \equiv [(P(x, y) \wedge P(y, z)) \rightarrow P(x, z)] \quad (P \text{ is} \\
& & \text{transitive.)} \\
\phi_3 &\equiv & \forall x \; \forall y \; \eta_3, \quad \eta_3 \equiv \neg P(x, x) \wedge [(P(x, y) \rightarrow \neg P(y, x)] \\
& & (P \text{ is neither reflexive nor symmetric.)} \\
\phi_4 &\equiv & \forall x \; \forall y \; \eta_4, \quad \eta_4 \equiv S(x, y) \rightarrow [G(x, y) \wedge P(x, y)] \quad \text{(If an edge is} \\
& & \text{chosen by } S, \text{ then it has to be in the given graph } G \text{ and in the} \\
& & s - t \text{ path } P.) \\
\phi_5 &\equiv & \forall x \; \forall y \; \hat{\eta}_5 \text{ with } \hat{\eta}_5 \equiv P(x, y) \rightarrow [S(x, y) \vee \exists z(P(x, z) \wedge S(z, y))], \\
& & \text{(Recursive definition of } P \text{ — either there is an } (x, y) \text{ arc, or} \\
& & \text{there exists a path from } x \text{ to } z \text{ and a } (z, y) \text{ arc.)} \\
\phi_6 &\equiv & \forall x \; \forall y \; \forall z \; \eta_6, \quad \eta_6 \equiv [(S(x, y) \wedge S(z, y)) \rightarrow (x = z)] \quad \text{(Predecessor} \\
& & \text{is unique, hence there is a unique path } P \text{ from } s \text{ to } t.)
\end{aligned}
$$

It can be shown that each η_i $(1 \le i \le 6)$ above is equivalent to a Horn clause — clauses with at most one positive literal from the set of second order variables[6] $\{P, S\}$ — or a conjunction of such clauses, as required by Theorems 1 and 6. The optimal solution value for instance **G** can now be written in Horn Π_1 form as

$$opt(\mathbf{G}) = \min_{P,S} \left| \left\{ (p, q) : (\mathbf{G}, P, S) \models \; \forall x \; \forall y \; \forall z \; \bigwedge_{i=1}^{6} \eta_i \wedge S(p, q) \right\} \right|. \tag{25}$$

Discussion. Though we have not proved it, SHORTEST PATH$_{PB}$ could be one of those problems where the decision version can be represented in Σ_1 form, but the optimisation problem is in Π_1 form. It would be interesting if this observation (hierarchy in terms of quantifier complexity) could be proven or disproven.

---

[6]A clause such as $P(x, z) \vee S(s, t)$ cannot be a Horn clause, for instance.

3 Optimisation Problems in P′

We next turn our attention to the class $\mathbf{P}'$. This is the set of all optimisation problems, *not necessarily polynomially bound*, but the optimal solution can be computed within time polynomial in the size of the input. (These problems need not obey Equation 2.)

Zimand 1998 [Zim98] generalised Theorems 2 and 5 to all NP-Optimisation problems, not just those that are polynomially bound. He showed that a Π_2 first-order formula captures the feasibility conditions for any problem in this class, while the optimal solution value can be represented by a maximisation (or minimisation) over weighted tuples — the tuples are similar to those used in expressions (11) and (23) for polynomially bound problems, but now they are also assigned real number weights. The method of attaching weights to tuples has also been discussed in Papadimitriou and Yannakakis 1991 [PY91].

We will demonstrate (without a formal proof) that Zimand's result can be extended to polynomial-time maximisation problems as well. Zimand shows that for any positive integer value z for an optimal solution, we can compute a set of weights c_i that are powers of two, such that $z = \sum_i c_i$. However, for a *given* optimisation problem Q', the weights on tuples are *given quantities*, such as the arc capacities in a MAXFLOW problem or the arc costs in a TRAVELLING SALESPERSON problem — the weights are part of the input. (Zimand makes no attempt to relate his computed weights with the input weights.)

Grädel's Theorem states that any decision problem $Q \in \mathbf{P}$ can be represented as

$$Q \equiv \exists \mathcal{S} \psi. \tag{26}$$

A decision version of a maximisation problem asks if there is a solution $\mathcal{S}$ to an instance I (represented by a structure $\mathbf{A}$) such that the objective function $f(\mathbf{A}, \mathcal{S}) \geq K$ where K is a given constant.

Motivation to attach weights to tuples. If I is a YES instance to Q, then ψ must be able to express the fact that $f(\mathbf{A}, \mathcal{S}) \geq K$ using a finite structure, according to Grädel's Theorem. The quantity $f(\mathbf{A}, \mathcal{S})$, though not polynomially bound in the size of I any more, is still a finite quantity. For a structure $\mathbf{A}$ with universe A, the number of $k-$ary tuples possible is $|A|^k$, which is polynomial in the size of the instance. In other words, $f(\mathbf{A}, \mathcal{S})$ need not be polynomially bound, whereas the maximum number of $\mathbf{w}$ tuples should be — this is in contrast to the problems in Sect. 2. One way to capture a larger number ($f(\mathbf{A}, \mathcal{S})$) using a smaller one (the number of $\mathbf{w}$ tuples) is by attaching weights to the tuples.

For example, in the MIN CUT problem (dual of MAX FLOW), the tuples (arcs) are binary, and the weights of these tuples are the arc capacities. In WEIGHTED MAX3SAT, the tuples (clauses) are ternary with a weight attached to each clause. In WEIGHTED MAXSAT, it is unknown how many literals are in each clause, hence a unary tuple is commonly used [KT94]. In TRAVELLING SALESPERSON (TSP), the weights on the binary tuples are the arc costs. It may be undecided ahead of time how the optimal value to a problem in $\mathbf{P}'$ can be represented, as to which set of tuples and their weights will be used — for example, the set of edges used in a solution to the TSP is unknown until a solution is determined. However, the number of such sets and their tuples are finite — and the weight of each tuple is a given quantity.

Naturally, each set of tuples described above can be said to form a relation U_i over the universe of $\mathbf{A}$, and the set of all such relations can be represented by $\mathbf{U}$. For $U_i \in \mathbf{U}$, its weight $w(U_i)$ is defined as

$$w(U_i) = \sum_{\mathbf{w} \in U_i} w(\mathbf{w}), \tag{27}$$

where $w(\mathbf{w})$ is the given weight of tuple $\mathbf{w}$. For example, in a TSP instance with five vertices, each U_i will contain five tuples (the five arcs in the solution). However, in SHORTEST PATH, the number of tuples in U_i depends on the path (solution) used — hence the cardinality of the different U_i's is not the same, since the number of arcs in each solution can vary.

Furthermore, the universe A should consist of values (such as vertex indices in graphs) for the variables, as well as weights for the tuples[7]. A unary relation $C(x)$ — sometimes known as a *hidden relation* — decides if a given variable is a *basic* variable, or a weight for one of the tuples. The universe A of a structure $\mathbf{A}$ will be of the form

$$A = \{a_1, a_2, \cdots a_n, w_1, w_2, \cdots w_m\} \tag{28}$$

where the a_i's are possible values for the basic variables and the w_j's are possible weights for tuples of basic variables. For any variable x_i, the following expression $\phi_1(x_i)$ should hold:

$$\phi_1(x_i) \equiv [C(x_i)] \longleftrightarrow \bigvee_{j=1}^{n} [x_i = a_j]. \tag{29}$$

Introduce a relation $R(x_0, x_1, x_2, \cdots x_k)$ which holds true iff x_0 is a weight for the tuple $\mathbf{w} = (x_1, x_2, \cdots x_k)$ — hence $C(x_0)$ is false, and all other $C(x_i)$'s are true. In an instance, the variable x_0 is instantiated with a weight w_i from A.

3.1   Maximisation Problems

Reverting to Grädel's expressibility in (26), since ψ is in Π_1 ESO Horn form, it can be written as (just like the case for polynomially bound problems),

$$\psi(\mathbf{x}, \mathbf{w_i}, \mathcal{S}) = \forall x_1 \forall x_2 \cdots \forall x_m \; \hat{\eta}(x_1, x_2, \cdots x_m, \mathbf{w_i}, \mathcal{S}) = \forall \mathbf{x} \; \hat{\eta}(\mathbf{x}, \mathbf{w_i}, \mathcal{S}) \tag{30}$$

where $\mathbf{x} = (x_1, x_2, \cdots x_m)$ — hence $\hat{\eta}$ should express the fact that $w(U_i) \geq K$, and $\hat{\eta}$ should include expressions for every $\phi_1(x_i)$ in (29). If a certain relation U_i that satisfies the feasibility conditions exists, then

$$Q \equiv \exists \mathcal{S} \; \exists U_i \; \forall \mathbf{w} \; \forall \mathbf{x} \, [U_i(\mathbf{w}) \longleftrightarrow \hat{\eta}(\mathbf{x}, \mathbf{w}, \mathcal{S})] \,, \tag{31}$$

where Q is the decision problem. From this, the value of the optimal solution (for the optimisation problem Q') can be expressed as

$$opt_{Q'}(\mathbf{A}) = \max_{\mathcal{S}, U_i} \{w(U_i) : (\mathbf{A}, \mathcal{S}, U_i) \models \forall \mathbf{w} \forall \mathbf{x} \, [U_i(\mathbf{w}) \longleftrightarrow \hat{\eta}(\mathbf{x}, \mathbf{w}, \mathcal{S})]\} \,. \tag{32}$$

$(\mathbf{A}, \mathcal{S}, U_i)$ above also satisifies expressions where $U_i(\mathbf{w})$ and $\hat{\eta}(\mathbf{x}, \mathbf{w}, \mathcal{S})$ are false — however, since $U_i(\mathbf{w})$ is false, the weight of this tuple $\mathbf{w}$ will not be counted in $w(U_i)$.

Note that (32) need not be a Π_1 Horn formula any more, since the Horn property of $\neg \hat{\eta}(\mathbf{x}, \mathbf{w}, \mathcal{S})$ is unknown:

$$[U_i(\mathbf{w}) \longleftrightarrow \hat{\eta}(\mathbf{x}, \mathbf{w}, \mathcal{S})] \equiv [U_i(\mathbf{w}) \vee \neg \hat{\eta}(\mathbf{x}, \mathbf{w}, \mathcal{S})] \wedge [\neg U_i(\mathbf{w}) \vee \hat{\eta}(\mathbf{x}, \mathbf{w}, \mathcal{S})] \,. \tag{33}$$

---

[7]This is a variant of Many-sorted Logic.

3.2 Example: Weighted Matching

Here, we provide an example of how WEIGHTED MATCHING (optimisation version) can be expressed. Given a graph **G** with weights on the edges, the objective is to *mark* certain edges such that the sum of the weights on the marked edges is maximised, with the condition that no two adjacent edges in **G** can be marked. (In the context of this problem, a *Matched edge* is a synonym for a *Marked edge*.) An instance (structure) **A** consists of
(a) the universe A (the union of the set of vertices and the set of tuple-weights),
(b) a relation G (the set of edges),
(c) a relation $C(x)$, which defines whether a variable is a vertex or the weight of a tuple,
(d) and a ternary relation $R(x_0, x_1, x_2)$ that decides whether an edge (x_1, x_2) is assigned a weight of x_0.

Let relation $U(v_i, v_j)$ be true if (v_i, v_j) is a matched edge. Obviously it can be a matched edge only if the edge exists in the given graph. This is expressed by ϕ_0 below. If edge (v_i, v_j) is matched and x is a vertex not in $\{v_i, v_j\}$, then an adjacent edge $G(x, v_i)$ (if it exists in the given graph) cannot be matched. This is expressed by ϕ_1. The three other expressions ϕ_2, ϕ_3 and ϕ_4 perform the same task.

$$\phi_1 = U(v_i, v_j) \rightarrow G(v_i, v_j),$$

$$\tau = (x \neq v_i) \wedge (x \neq v_j) \wedge U(v_i, v_j),$$

$$\phi_1 = \tau \wedge G(x, v_i) \rightarrow \neg U(x, v_i), \quad \phi_2 = \tau \wedge G(v_i, x) \rightarrow \neg U(v_i, x),$$

$$\phi_3 = \tau \wedge G(x, v_j) \rightarrow \neg U(x, v_j), \quad \phi_4 = \tau \wedge G(v_j, x) \rightarrow \neg U(v_j, x).$$

Let set of weights $B = \{z \in A \mid \exists x \exists y\, U(x, y) \wedge R(z, x, y)\}$ — however, since B is a set, if the same weight is assigned to two or more edges in U, only one of them will be counted towards total edge weights. Thus there is a need to split B into B_i ($1 \leq i \leq m$, $m =$ number of edges in the input) — a weight w in B_i occurs among i edges in U. Hence[8]

$$B_1 \quad = \quad \{z \in A \mid \exists x\, \exists y\, \forall u\, \forall v\, \tau \wedge U(x, y) \wedge R(z, x, y)\}, \tag{34}$$

$$\text{where} \quad \tau \quad = \quad \{[(u \neq x) \vee (v \neq y)] \wedge U(u, v)\} \rightarrow \neg R(z, u, v). \tag{35}$$

(Note: In the definition of B_k below, $\exists_{i=1}^k x_i$ is a shorthand for $\exists x_1\, \exists x_2 \cdots \exists x_k$.)

In general, any B_k ($1 \leq k \leq m$) can be expressed as

$$B_k = \left\{ z \in A \mid \exists_{i=1}^k x_i\, \exists_{i=1}^k y_i\, \forall u\, \forall v\, \bigwedge_{i=1}^k U(x_i, y_i) \bigwedge_{i=1}^k R(z, x_i, y_i) \wedge \tau \right\}, \tag{36}$$

where $\tau = \tau_1 \wedge \tau_2$, and,

$$\tau_1 = \left\{ \bigwedge_{i=1}^k [(u, v) \neq (x_i, y_i)] \wedge U(u, v) \right\} \rightarrow \neg R(z, u, v), \tag{37}$$

$$\tau_2 = \bigwedge_{i \neq j} (x_i, y_i) \neq (x_j, y_j) \tag{38}$$

---

[8]Issues such as quantifier complexity and Horn property are irrelevant for the logic expressions in (34)-(39). Logic expressions are used here for the sole purpose of defining B_k, $1 \leq k \leq m$.

$$\{(x_i, y_i) \neq (x_j, y_j)\} \equiv \{(x_i \neq x_j) \vee (y_i \neq y_j)\}. \tag{39}$$

Expression (38) says that there are k distinct edges (x_i, y_i). Expression (39) is an explanation of the shorthand notation used in (37) and (38) — that if two edges are different, then at least one of their endpoints should be different.

The weight of relation U, $w(U)$, is computed as:

$$w(U) = \left(\sum_{z \in B_1} z \right) + \left(2 \sum_{z \in B_2} z \right) + \cdots + \left(m \sum_{z \in B_m} z \right). \tag{40}$$

Finally, Φ is the expression that a solution U should satisfy, and the optimal solution value is obtained by maximising over all such solutions:

$$\Phi = \forall v_i \, \forall v_j \, \forall x \, [C(v_i) \wedge C(v_j) \wedge C(x)] \rightarrow \bigwedge_{k=0}^{4} \phi_k, \tag{41}$$

$$opt_{Q'}(\mathbf{A}) = \max_U \{w(U) : (\mathbf{A}, U) \models \Phi\}. \tag{42}$$

4   Future Research

The open question — mentioned in the proof to Theorem 3 — of how to express decision versions of optimisation problems in the Π_1 form specified by Grädel for problems in $\mathbf{P}$ in Sect. 2.1 needs resolution. A formal proof is needed for the arguments in Sect. 3.1. Furthermore, Sect. 3 studies only maximisation problems — research should be carried out for minimisation problems as well. Complete problems should be discovered for the respective subclasses.

Since the decision version of the weighted MAXFLOW problem (where arc capacity can be any non-negative integer) is complete for the class $\mathbf{P}$ [GHR95, Imm99], the optimisation version of weighted MAXFLOW is likely to be a complete problem for $\mathbf{P}'$ — this is yet to be proven.

Acknowledgements

References

[E. 91]   E. Grädel. The expressive power of second order Horn logic. In *STACS 1991: Proceedings of the 8th annual symposium on Theoretical aspects of computer science — Lecture Notes in Computer Science 280*, pages 466–477. Springer-Verlag, 1991.

[Fag74] R. Fagin. Generalized first-order spectra and polynomial-time recognizable sets. In R. Karp, editor, *Complexity of Computations*, pages 43–73. SIAM-AMS Proceedings (no.7), 1974.

[GHR95] R. Greenlaw, H. James Hoover, and W.L. Ruzzo. *Limits to Parallel Computation: P-Completeness Theory*. Oxford University Press, 1995.

[GJ79] M.R. Garey and D.S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman (New York), 1979.

[H97] J. Håstad. Some Optimal Inapproximability Results. In *ACM-STOC 1997: Proceedings of the 29th ACM Symposium on the Theory of Computing*, pages 1–10, 1997.

[Imm99] Neil Immerman. *Descriptive Complexity*. Springer-Verlag, 1999.

[KKM94] R. Kohli, R. Krishnamurti, and P. Mirchandani. The Minimum Satisfiability Problem. *SIAM Journal of Discrete Mathematics*, 7:275–283, 1994.

[KMSV98] S. Khanna, R. Motwani, M. Sudan, and U. Vazirani. On syntactic versus computational views of approximability. *SIAM Journal of Computing*, 28(1):164–191, 1998.

[KT94] P.G. Kolaitis and M.N. Thakur. Logical Definability of NP-Optimisation Problems. *Information and Computation*, 115(2):321–353, December 1994.

[KT95] P.G. Kolaitis and M.N. Thakur. Approximation Properties of NP-Minimisation Problems. *Journal of Computer and System Sciences*, 50:391–411, 1995.

[Pap94] C.H. Papadimitriou. *Computational Complexity*. Addison-Wesley (Reading, Massachusetts), 1994.

[PR93] A. Panconesi and D. Ranjan. Quantifiers and approximation. *Theoretical Computer Science*, 107:145–163, 1993.

[PY91] C.H. Papadimitriou and M. Yannakakis. Optimization, Approximation, and Complexity Classes. *Journal of Computer and System Sciences*, 43(3):425–440, December 1991.

[Zim98] M. Zimand. Weighted NP-Optimisation Problems: Logical Definability and Approximation Properties. *SIAM Journal of Computing*, 28(1):36–56, 1998.